

Klausuren und der Aufgabenpool – Kurzanleitung

Dieses Dokument beschreibt, wie Klausuren mit dem offiziellen **Klausurtemplate** des AIFB erstellt sowie der dazu gehörende Aufgabenpool benutzt werden können. Die neue Version hat viele nützliche Features und wird im SVN archiviert; **alle älteren Versionen (auch die Standalone-Version aus dem Sharepoint) sollten nicht mehr verwendet werden.**

Schnellstart

0. Vorbereitung:

- a) Wer es noch nicht hat, miktex installieren: <http://miktex.org>
- b) Einen Editor installieren, bspw.: <http://www.texniccenter.org>
- c) Tortoise SVN installieren: <http://tortoisesvn.net>
- d) Bei Kaibin Schreib- und Leserechte für dieses Verzeichnis beantragen:

```
https://aifburanos.aifb.kit.edu/svn/lehre/aufgabenpool
```

Dann dieses Verzeichnis auschecken.

1. Nach dem Auschecken:

- a) Die Datei `TeX/Pooldaten-Vorlage.txt` in `TeX/Pooldaten.tex` umkopieren (nicht verschieben!).
- b) Jetzt können in der neuen Datei die gewünschten Einstellungen zum Erzeugen von Klausuren, Aufgabenblättern usw. vorgenommen werden.

(Sonst muss im TeX-Verzeichnis direkt nichts verändert werden.)

2. Klausuren erstellen:

- a) Eine neue Klausur kann am einfachsten erstellt werden, indem unter `TeX/Klausuren` eines der schon vorhandenen Verzeichnisse in ein neues Verzeichnis `*myDir*` kopiert und angepasst wird.
- b) Die Klausur wird eingebunden, indem das neue Verzeichnis in der Datei `TeX/Pooldaten.tex` an folgender Stelle eingetragen wird:

```
def\klausur{./Klausuren/*myDir*}
```

In dieser Datei kann auch über die Boolesche Variable `mitloesung` gesteuert werden, ob die Klausur mit oder ohne Lösungen erzeugt wird.

- c) Im Verzeichnis `*myDir*` dürft ihr machen, was ihr wollt. Klausurdaten und Aufgaben bearbeiten, Aufgaben löschen, hinzufügen usw. – es sollte eigentlich intuitiv sein. In Abschnitt „Nützliche Umgebungen und Makros“ (s. u.) wird erklärt, wie die Aufgaben-, Lösungs-, Bewertungs-, ... Umgebungen u. ä. funktionieren.

3. Kompilieren: Kompiliert wird immer die (nicht zu modifizierende) Hauptdatei `PoolTemplate.tex`

4. **Committen: Eure neuen Klausuren/Aufgabenblätter etc. dürft und SOLLT ihr ins SVN committen, damit sie für spätere Generationen archiviert werden.**

5. Die Batch-Dateien im obersten Verzeichnis sind Hilfs-Methoden, die bisher vor allem für Info II die gewünschten Dokumente automatisch erzeugen. Man darf sie gerne modifizieren, sodass sie auch für die anderen Veranstaltungen funktionieren. (Wenn man Lukas lieb fragt, macht er das auch gerne :-)

Nützliche Umgebungen und Makros

Im Folgenden wird die Verwendung einiger wichtiger vorimplementierter Makros erklärt. Ein vorangestelltes κ steht für Makros, die **beim Erstellen von Klausuren** notwendig sind. Ein $\kappa!$ steht vor Makros, die **ausschließlich in Klausuren** verwendet werden können.

- $\kappa!$ • Das Makro `\klausur` zeigt immer genau auf das **Verzeichnis der aktuellen Klausur**. Wenn darin etwa eine Grafik `test.pdf` liegt, die man einbinden möchte, schreibt man einfach

```
\includegraphics{\klausur/test.pdf}
```

Sollte das Verzeichnis später einmal umbenannt werden, funktioniert der Befehl dann trotzdem weiterhin ohne Veränderung.

- κ • Die zentrale Umgebung, um **Aufgaben** einzubinden, ist:

```
\begin{aufgabe}[*Punkte*]{*Thema*}{*Schwere*}{*URL*}{*ID*}
  Aufgabentext
\end{aufgabe}
```

Außerhalb von Info II braucht man **meistens nur den ersten nichtoptionalen Parameter „Thema“**. Hier gibt man einen schönen Titel für die Aufgabe ein. Alle übrigen Parameter lässt man in diesem Fall leer. Der optionale Parameter „Punkte“ wird in Klausuren genutzt, **aber nur, wenn eine Aufgabe keine Teilaufgaben hat**, siehe nächster Punkt; sonst wird der Parameter einfach weg- oder leergelassen.

- κ! • In Klausuren kann man innerhalb einer Aufgabe **Teilaufgaben** definieren, meistens als Teil einer „enumerate“-Umgebung:

```
\begin{enumerate}
  \item \begin{teilaufgabe}{4}
    Hier kommt der Aufgabentext.
  \end{teilaufgabe}
\end{enumerate}
```

Die Punkte (im Beispiel 4) werden in einen Kasten am rechten Seitenrand geschrieben und mit den übrigen Punkten der Klausur verrechnet. Ganz oben, unterhalb der Aufgabendefinition, wird ein weiterer Kasten mit der Punktesumme für diese Aufgabe angezeigt.

/ 4

Wichtig: Beim Nutzen dieser Umgebung darf auf keinen Fall zusätzlich der optionale Parameter der aufgabe-Umgebung genutzt werden, sonst stimmt die Summe nicht!

- κ • Wenn ein Text oder eine Grafik **nur in der Lösung** angezeigt werden soll, wenn also in Pooldaten.tex mitloesung auf true gesetzt ist, kann die folgenden Umgebung verwendet werden:

```
\begin{loesung} *Lösungstext* \end{loesung}
```

- κ • Das gleiche gibt es umgekehrt, wenn mitloesung auf false gesetzt ist:

```
\begin{nichtloesung} *Nichtlösungs-Text* \end{nichtloesung}
```

- Um ein **einheitliches Format** zu wahren, kann die Hilfs Umgebung loesungplain verwendet werden, die für eine einheitliche Überschrift „**Lösung:**“ und ausreichend Abstände zwischen vorhergehendem und darauffolgendem Text sorgt und automatisch zwischen den Sprachen Englisch und Deutsch umschaltet, je nachdem, welche Sprache in klausurdaten.tex eingestellt ist.

```
\begin{loesung}\begin{loesungplain} *Lösungstext* \end{loesungplain}\end{loesung}
```

- κ! • Bei Klausuren gibt es auch eine **Bewertungsumgebung**, die nur angezeigt wird, wenn in eurem Klausurverzeichnis *mydir* in der Datei klausurdaten.tex der Wert mitbewertung auf true gesetzt ist:

```
\begin{bewertung} *Bewertungstext* \end{bewertung}
```

- Der **XWizard** kann dazu genutzt werden, um Konzepte aus den Vorlesungen **Info II, Effiziente Algorithmen und AIA** im Web darzustellen. Auf die Objekte können Algorithmen angewendet und die Resultate können als PDF heruntergeladen werden. Die zu den Objekten gehörenden Skripte werden in einer Datenbank gespeichert und können per Short-URL aufgerufen werden können. Für diese Short-URLs lassen sich QR-Codes erzeugt werden, die in Dokumenten des Aufgabenpools angezeigt werden können. Beispiel:

Führt zu folgendem Link mit QR-Code:

SKRIPT ID-9743



Klickt man ihn an, gelangt man zu einer Chomsky-Grammatik, die man mit verschiedenen Konversionsmethoden bearbeiten kann. Ganz unten auf der Web-Seite gibt es Beispiele zu den verfügbaren Objekt-Typen.

- **Dreieckstabellen** sind in \LaTeX etwas heikel zu erstellen. Daher haben wir einmal Zeit investiert, um das Problem durch zwei Makros ein für alle mal zu lösen. Das erste Makro heißt `\dreieckstabelle` und funktioniert so:

\dreieckstabelle

```

{\$s_1\$ & \$\times_{0}\$}
{\$s_2\$ & \$\times_{0}\$ & \$-\$}
{\$s_3\$ & \$\times_{1}\$ & \$\times_{0}\$ & \$\times_{0}\$}
{\$s_4\$ & \$\times_{0}\$ & \$-\$ & \$-\$ & \$\times_{0}\$}
{\$s_5\$ & \$\times_{0}\$ & \$\times_{2}\$ & \$\times_{2}\$ & \$\times_{0}\$ & \$-\$}
{ & \$s_{0}\$ & \$s_{1}\$ & \$s_{2}\$ & \$s_{3}\$ & \$s_{4}\$}
{}{}{}{}{}{}{}{}{}{}{}{}

```

Das Ergebnis ist:

s_1	\times_0				
s_2	\times_0	$-$			
s_3	\times_1	\times_0	\times_0		
s_4	\times_0	$-$	$-$	\times_0	
s_5	\times_0	\times_2	\times_2	\times_0	$-$
	s_0	s_1	s_2	s_3	s_4

Das zweite Makro heißt `dreieckstabelleReverse` und funktioniert folgendermaßen (die Leerzeichen sind hier natürlich nur zur besseren Lesbarkeit der Spalten eingefügt und können weggelassen werden):

\dreieckstabelleReverse

```
{          & $b$          & $b$          & $a$          & $a$          & $a$          & $b$}
{$m=1$ & $B$          & $B$          & $A,C$        & $A,C$        & $A,C$        & $B$}
{$m=2$ & $-$          & $S,A$        & $B$          & $B$          & $S,C$}
{$m=3$ & $A$          & $-$          & $S,A,C$      & $B$}
{$m=4$ & $-$          & $S,A,C$      & $S,C$}
{$m=5$ & $S,A,C$      & $S,C$}
{$m=6$ & $S,C$}
{ } { } { } { } { } { } { } { } { } { }
```

Das Ergebnis ist:

	b	b	a	a	a	b
$m = 1$	B	B	A, C	A, C	A, C	B
$m = 2$	$-$	S, A	B	B	S, C	
$m = 3$	A	$-$	S, A, C	B		
$m = 4$	$-$	S, A, C	S, C			
$m = 5$	S, A, C	S, C				
$m = 6$	S, C					

Man beachte, dass beide Makros **immer** genau 18 Parameter erhalten müssen, von denen nicht genutzte leer bleiben. (Wie Lukas es geschafft hat, mehr als neun Parameter zu übergeben, eigentlich T_EX's Maximum, dürft ihr selber ausknobeln oder in der Datei `aufgabenpool.sty` nachlesen.)

- Manchmal will man **Multiple-Choice-Fragen** (mit Wahr-Falsch-Auswahl) stellen. Auch dafür gibt es eine vorgefertigte Umgebung namens `mctabular`, in die man die einzelnen Fragestellungen durch das Makro `\mczeile` einfügt. Zu jeder Frage gibt man als Parameter zuerst (optional) eine **Erklärung** an, die nur in der Lösung angezeigt wird. Der zweite Parameter ist die **Frage** selber. Als dritten und vierten Parameter gibt man an, welche Lösung korrekt (`\bobbelX`) und welche falsch (`\bobbel`) ist (das wird natürlich auch nur in der Lösung angezeigt). Damit sowohl die Lösung als auch die Nicht-Lösung korrekt dargestellt werden, muss man aus technischen Gründen denselben Tabellentext zweimal schreiben, einmal innerhalb einer `nichtloesung`-Umgebung und einmal innerhalb einer `loesung`-Umgebung. Das folgende Beispiel verdeutlicht das Prinzip:

```
\begin{nichtloesung}
  \begin{mctabular} % Erste Kopie für Nicht-Lösung.
    \mczeile      % Erste Frage
      [Das Pumping-Lemma gilt für alle kontextfreien Sprachen.] % Erklärung
      {Nicht-kontextfr. Sprachen können das Pumping-Lemma erfüllen.} % Frage
      {\bobbelX} % Wahr
      {\bobbel} % Falsch
    \mczeile      % Zweite Frage
      [Beide Pumping-Lemmata werden meist in dieser Form verwendet.] % Erklärung
      {Man bedient sich eines Widerspruchsbeweis, um zu zeigen...} % Frage
      {\bobbelX} % Wahr
      {\bobbel} % Falsch
  \end{mctabular}
\end{nichtloesung}
%
\begin{loesung}
  \begin{mctabular} % Zweite Kopie für Lösung.
    \mczeile      % Erste Frage
      [Das Pumping-Lemma gilt für alle kontextfreien Sprachen.] % Erklärung
      {Nicht-kontextfr. Sprachen können das Pumping-Lemma erfüllen.} % Frage
      {\bobbelX} % Wahr
      {\bobbel} % Falsch
    \mczeile      % Zweite Frage
      [Beide Pumping-Lemmata werden meist in dieser Form verwendet.] % Erklärung
      {Man bedient sich eines Widerspruchsbeweis, um zu zeigen...} % Frage
      {\bobbelX} % Wahr
      {\bobbel} % Falsch
  \end{mctabular}
\end{loesung}
```

Das Ergebnis ist, wenn Lösungen angezeigt werden (wenn man die Lösungen ausschaltet, fehlen die Erklärungen und die gesetzten Kreuze):

	Wahr	Falsch
Es gibt nicht-kontextfreie Sprachen, die das Pumping-Lemma für kontextfreie Sprachen erfüllen.	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Erklärung: Die beiden Pumping-Lemmata sind Eigenschaften aller regulären bzw. kontextfreien Sprachen, allerdings nicht ausschließlich, denn es gibt auch nicht-reguläre bzw. nicht-kontextfreie Sprachen, die das jeweilige Pumping-Lemma erfüllen.

Man bedient sich des Pumping-Lemmas für kontextfreie Sprachen, um in einem Widerspruchsbeweis zu zeigen, dass eine Sprache nicht kontextfrei ist.	<input checked="" type="checkbox"/>	<input type="checkbox"/>
---	-------------------------------------	--------------------------

Erklärung: Beide Pumping-Lemmata werden meist in dieser Form verwendet.

Für den Einstieg empfiehlt es sich, schon vorhandene Aufgaben anzuschauen, bspw. aus einer alten Klausur. **Ansonsten stehen Lukas und das Info-II-Team auch immer gerne bei Fragen zur Verfügung.**

Umgebungen und Makros speziell für Info II

- **Endliche Automaten** werden durch folgendes Makro eingebunden (mit Definition und Zustandsübergangsdiagramm als PDF).

```
\myincludeFSM
[*size*]
{*PDF Überföhrungsfunktion*}
{*Bezeichner Überföhrungsfunktion*}
{*Automatendefinition*}
```

Beispielsweise (das Makro `\klausur` zeigt immer genau auf das aktuelle Verzeichnis der Klausur):

```
\myincludeFSM
[scale=\fsmScaleExam]
{\klausur/Quellen/automat.pdf}
{\delta}
{(\{a\}, \{s_0, s_1\}, \delta, s_0, \{s_0, s_1\})}
```

Zum Schluss noch ein paar rechtliche und strukturelle Dinge

1. Alles, was mit dem „Original-Aufgabenpool“ von Info II zu tun hat, ist kopierrechtlich etwas heikel und sollte nur nach Absprache mit Friederike oder Lukas an Dritte herausgegeben werden. Das gilt insbesondere für die Buch-gesamt.pdf in diesem Verzeichnis sowie für die entsprechenden Quelldateien.
2. Ansonsten ist aber von unserer Seite der gesamte Quellcode offen und kann gerne auch anderweitig verwendet und modifiziert werden. (Aber bzgl. der Kern-Dateien bitte nur außerhalb dieser SVN-Version des Aufgabenpools oder in Absprache mit dem Info-II-Team.) Neue Ideen oder Anregungen können gerne an uns weitergegeben werden.
3. Bitte committet keine Temp-Dateien vom Kompilierungsprozess. Die meisten sind schon auf die Ignore-Liste gesetzt, aber vermutlich nicht alle.

Viel Spaß mit dem Aufgabenpool!